An Open Source Computational Geometry Library for Parametric Aircraft Design

ESCO 2018 – Pilsen – Czech Republic

Martin Siggel
Simulation- and Software Technology
German Aerospace Center (DLR)

Outline

Introduction
- Aircraft design optimization
- TiGL Software overview

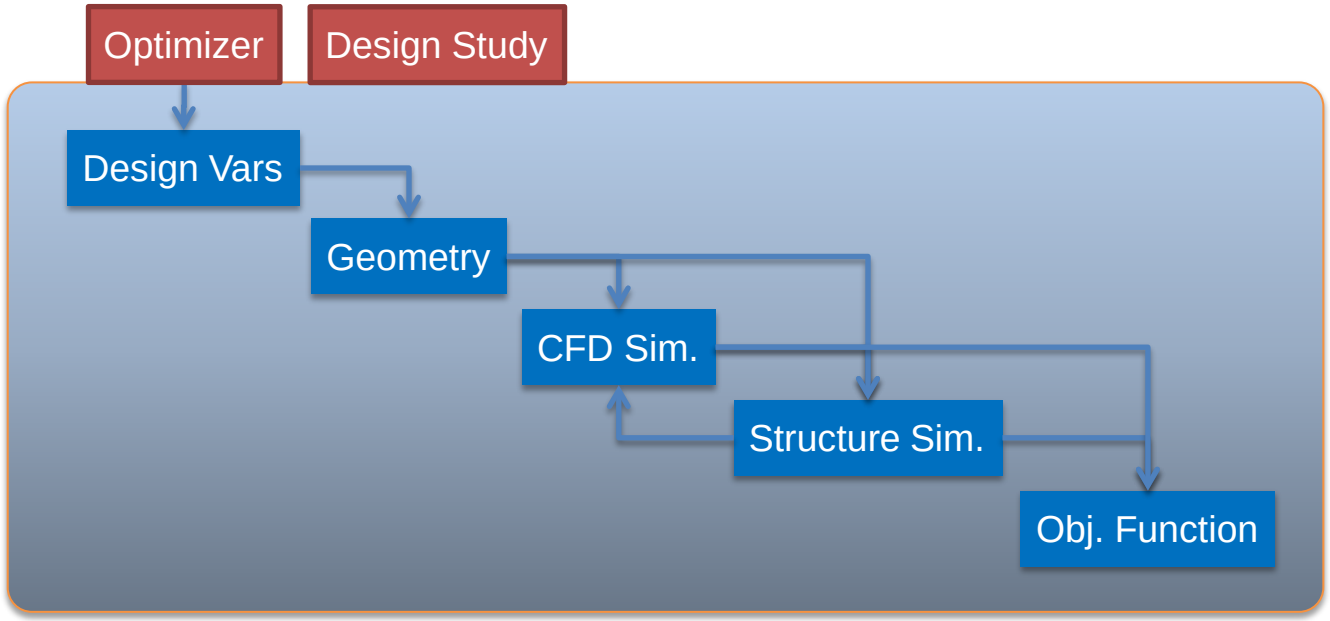
TiGL methods
- Applications and uses
- Architecture
- Curve and surface interpolation algorithms

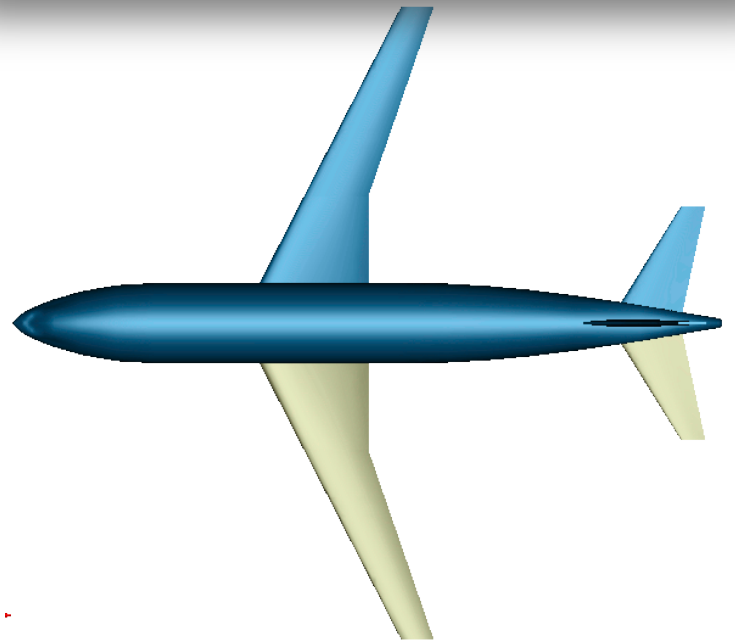
Results
- Comparison Gordon surfaces vs. Coons patches
- Complete aircraft geometries

Motivation

Aircraft design optimization overview

- Explore the aircraft of tomorrow
- Evaluate new designs

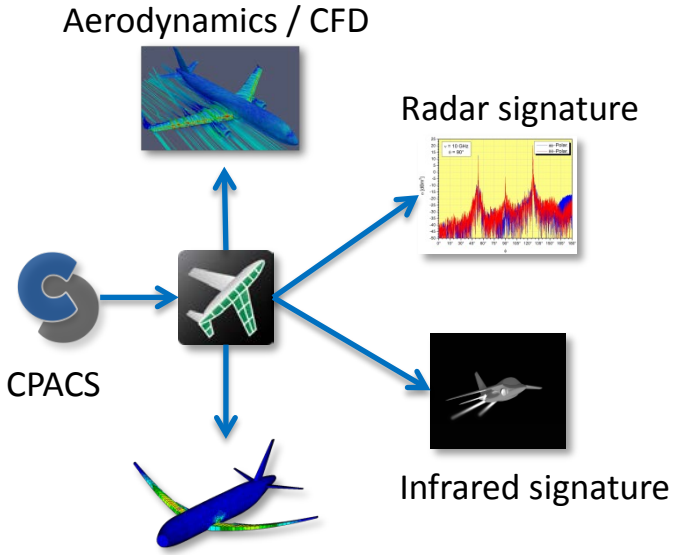


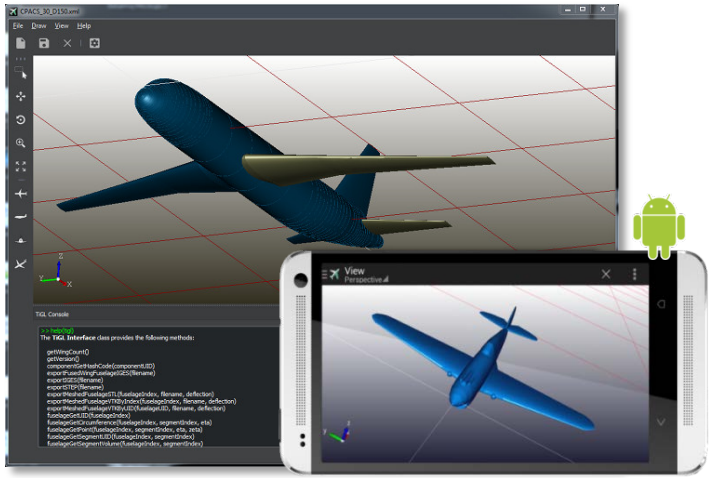


- Geometry generation has essential role
- All codes in the process chain must be robust

The TiGL software package

- C++ Library for parametric modelling of aircraft and helicopter based on parametric CPACS* (XML) files
- Used at DLR and international universities / research institutes for aircraft design and analysis
- TiGL Viewer to visualize CPACS-based aircraft geometries and other CAD files
- Cross platform: Linux, macOS, Windows, Android
- Open Source, developers from





*B. Nagel et. al., *Communication in Aircraft Design: Can we establish a Common Language?,* 28th ICAS, Brisbane, Australia, 2012

Introduction

- Aircraft design optimization
- TiGL Software overview

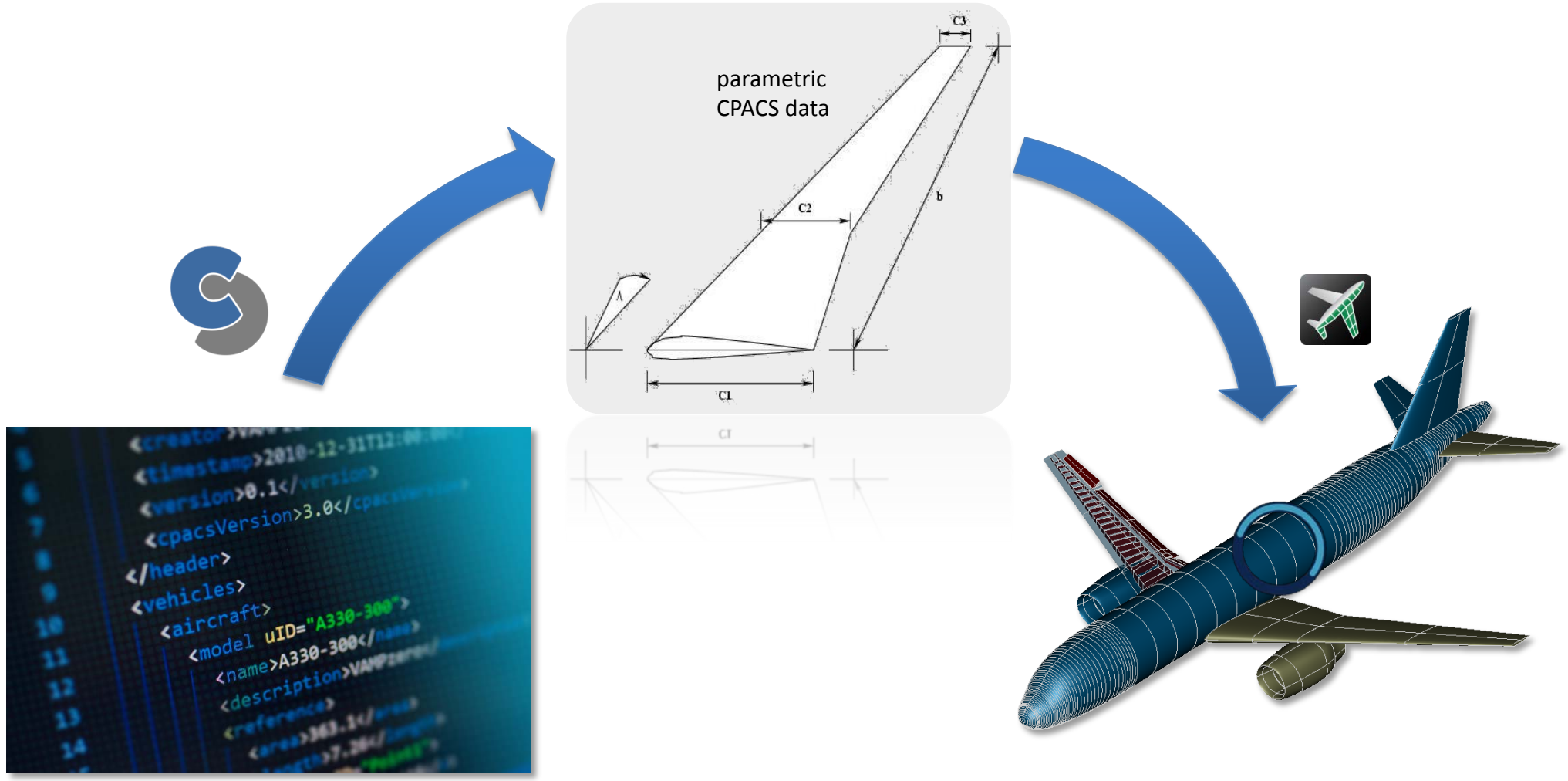
TiGL methods

- Features and applications
- Architecture
- Curve and surface interpolation algorithms

Results

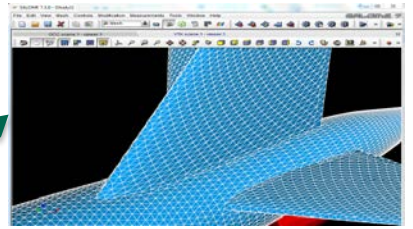
- Comparison Gordon surfaces vs. Coons patches
- Complete aircraft geometries

Parametric geometry

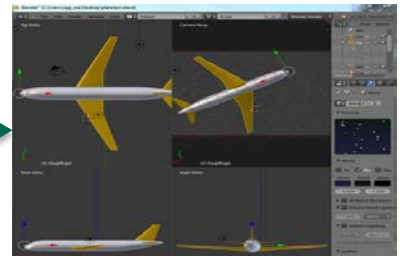


Features and Applications

- Custom geometric modeling algorithms
- Geometry export to common file formats, e.g.
  IGES, STEP, STL, VTK, Collada
- NURBS-based modelling of the main parts, e.g.
  - Wings
  - Fuselages
  - Engine covers
  - Wing structure
  - Flaps
  - Fuselage structure (at work)
- API to query
  - Points on surfaces
  - Intersections
  - Projections
- Mesh generation (at work)
  - Volume meshes for fluid dynamics simulations
  - Surface meshes for structural analysis, radar signatures

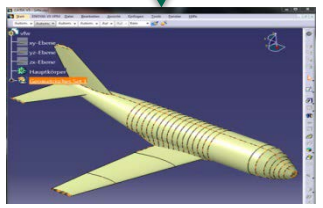


Mesh Generation for Simulations (at work)



Rendering, Visualization

3D Printing



Modeling with CAD Systems

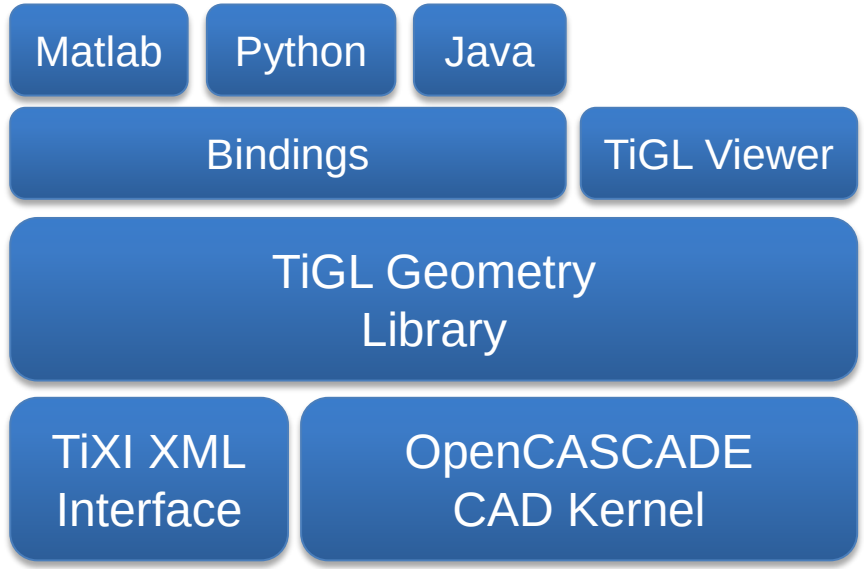
Architecture

- TiXI (https://github.com/dlr-sc/tixi)
  - Library to parse XML (CPACS) files

- OpenCASCADE (https://www.opencascade.com/)
  - Geometry (NURBS-based)
  - Topology (Boundary Representation)
  - CAD Exports, Visualization

- Language Bindings
  - Generated via SWIG (http://www.swig.org/)
  - Can access all C++ Data structures

- TiGL Viewer
  - 3D Visualization
  - Scripting
  - Debugging



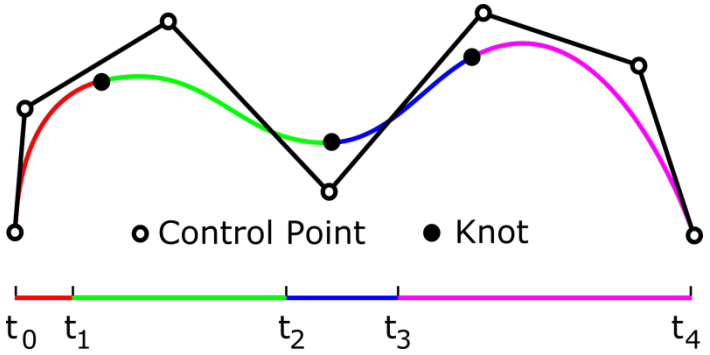
Under the hood
B-splines / NURBS

- B-spline curve:

$$\boldsymbol{c}(u) = \sum_{i=0}^{n} \boldsymbol{P}_i * N_i^d(u, \boldsymbol{t})$$

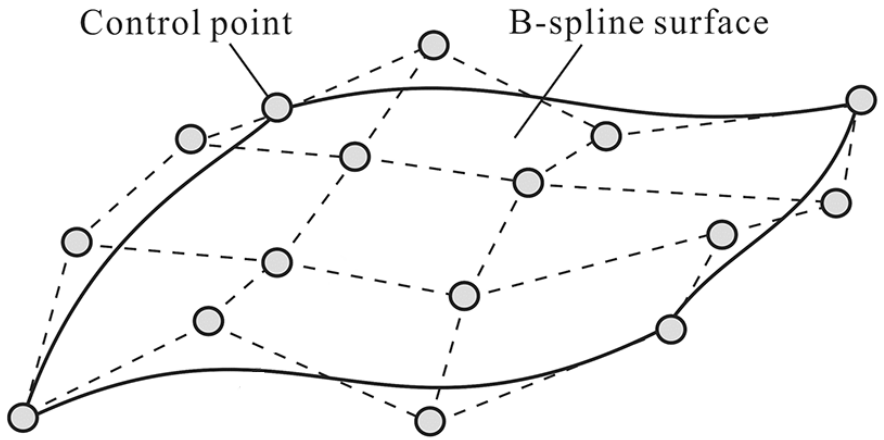
with:

- Control points $\{P_i^c\}$
- B-spline basis functions $N_i^d(u, \boldsymbol{t})$
- Knot vector $\boldsymbol{t}$, $t_i \leq t_{i+1}$



- B-spline surface:

$$\boldsymbol{s}(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} \boldsymbol{P}_{ij} * N_i^{d_u}(u, \boldsymbol{t}_u) * N_j^{d_v}(v, \boldsymbol{t}_v)$$



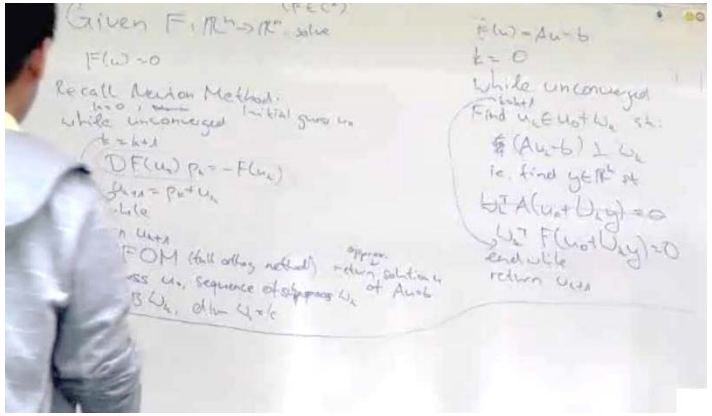
Under the hood
B-spline curve interpolation

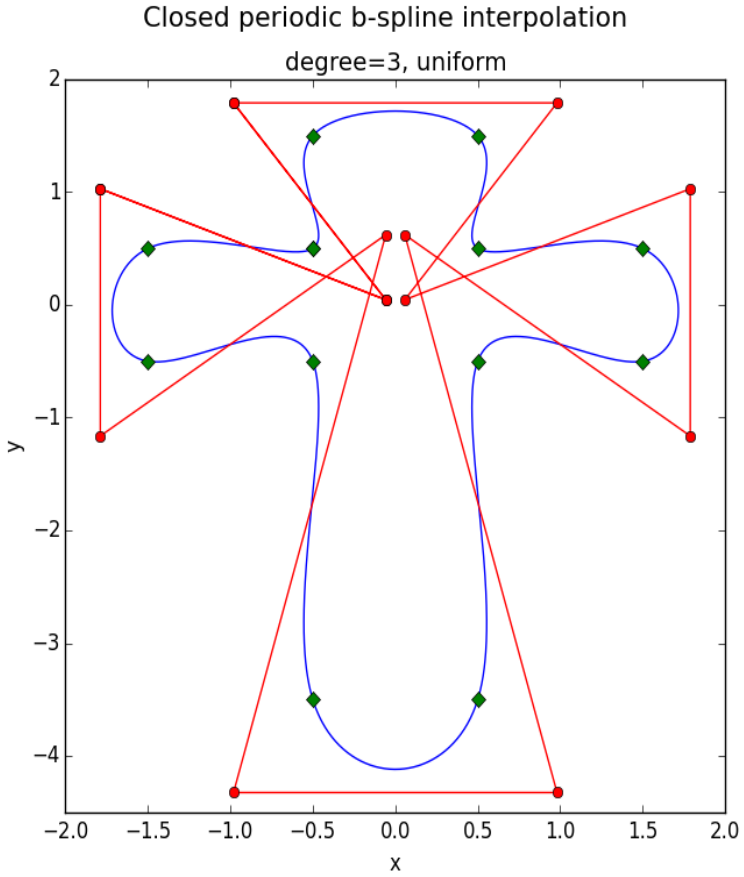
- Solve control points Pi, given data points Dj, such that:

$$\sum_{i=0}^{n} \boldsymbol{P}_i * N_i^d(u_j, \boldsymbol{t}) = D_j$$

$$\Rightarrow \boldsymbol{Np} \equiv \boldsymbol{d}$$

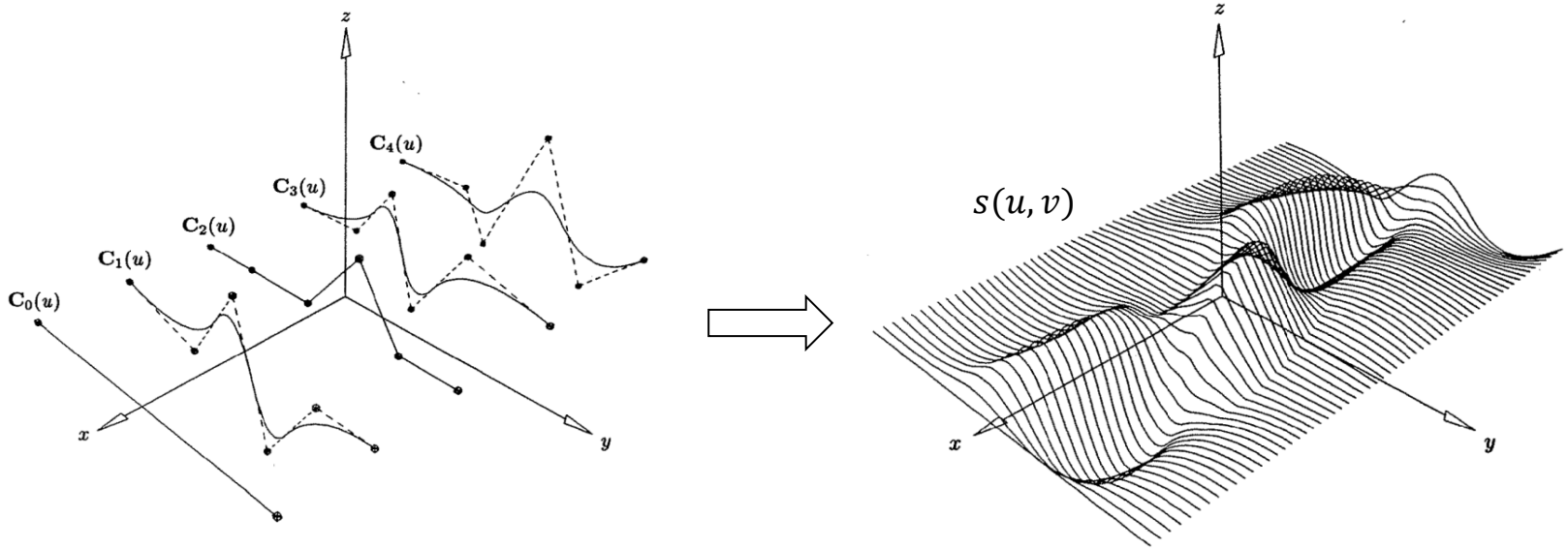
i.e. the curve passes though the data points





Under the hood
B-spline surface skinning

- Interpolates set of B-spline curves $c_i(u)$ by B-spline surface $s(u, v)$



- First: knot insertion to create same knot vector for all curves
- Then: Interpolate each row of control points with a curve → Control points of surface

Gordon Surfaces
Curve network interpolation

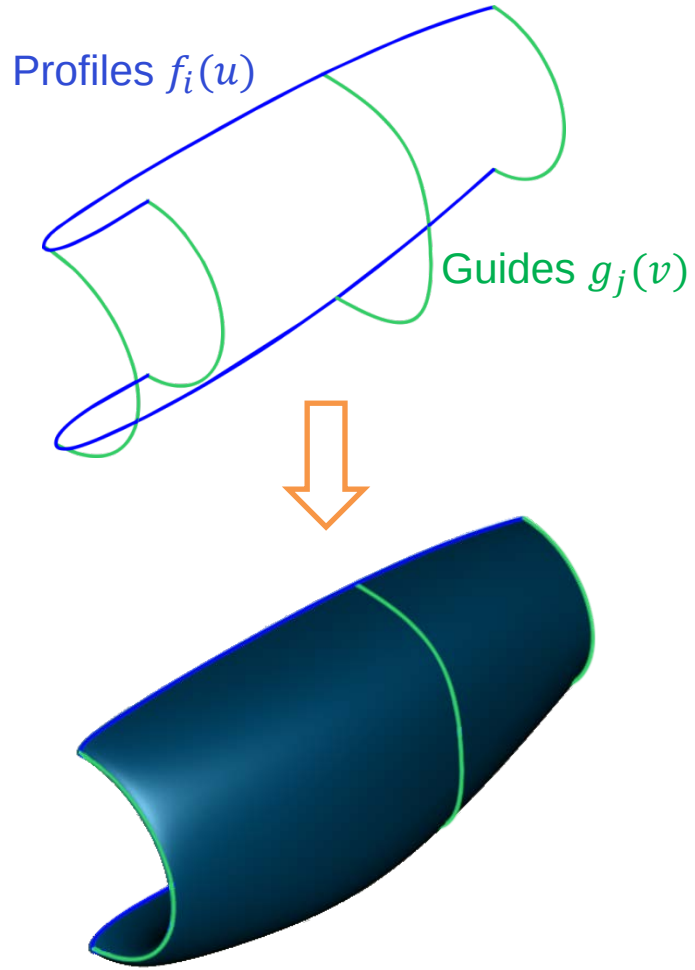
- Given network of profile and guide curves: Find surface that interpolates these curves
- Problem: No free library available for curve network interpolation!
- Custom development from OpenCASCADE for DLR based on Coons-patches showed poor results
- „Gordon Surfaces“ interpolate curve networks, but require the curves to be compatible

Compatibility condition

➢ All profile curves $f_i(u)$ must intersect with a guide curve $g_j(v)$at exactly the same parameter value u_j and vice versa:

$$f_i(u_j) = \mathrm{g_j}(\mathrm{v_i}), \forall i, j$$

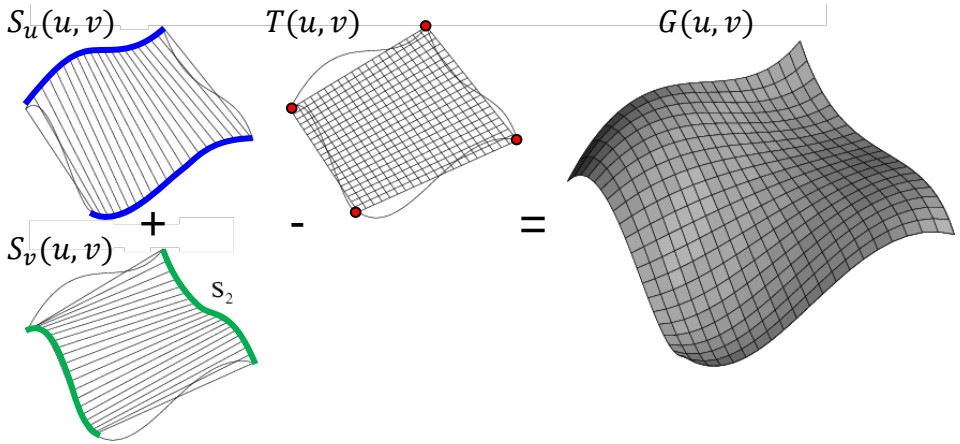
- Practically never the case → Reparametrization of the curves (tricky)



Gordon Surfaces
Algorithm overview

→ Gordon Surface is superposition of three surfaces:

$$G(u,v) = S_u(u,v) + S_v(u,v) - T(u,v)$$



$S_u(u,v)$: Skinning surface interpolating all profiles

$S_v(u,v)$: Skinning surface interpolating all guides

$T(u,v)$: Tensor product surface interpolating the intersection points of the curve network

→ Convert Gordon surface to B-Spline / NURBS for further use in TiGL
(degree elevation, knot insertion)

Gordon Surfaces
Robustness

- In theory: Profiles and guides must intersect each other!
- In practice: Allow for user defined tolerances
- Allow curve imperfections
- Reorder and reverse curves if necessary
- Try to handle ALL special cases!
- Provide Coons-based fallback solution

Introduction

- Aircraft design optimization
- TiGL Software overview

TiGL methods

- Applications and uses
- Architecture
- Curve and surface interpolation algorithms

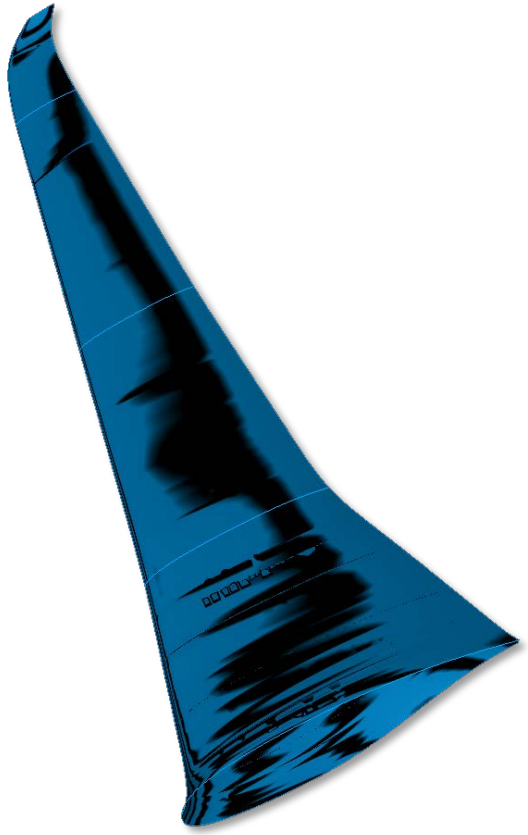
Results

- Comparison Gordon surfaces vs. Coons patches
- Complete aircraft geometries

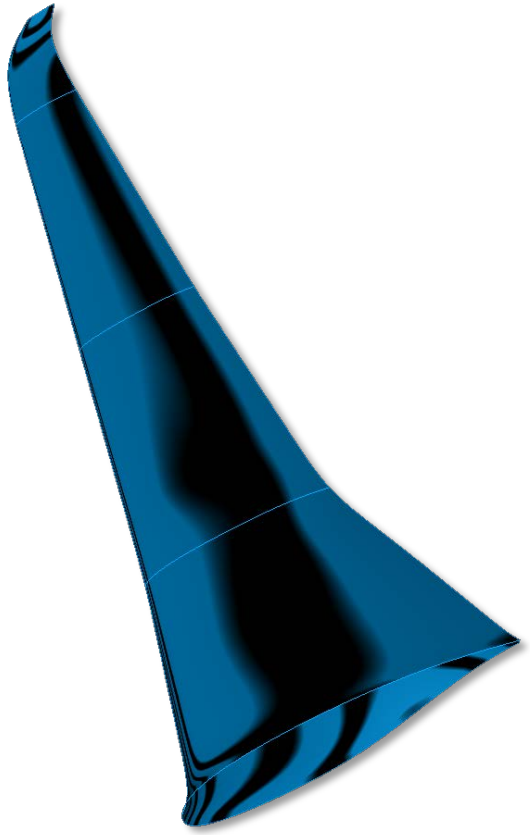
Surface Quality Analysis

- Surface quality analysis with zebra stripe plot
- Implemented in TiGL Viewer using OpenGL fragment shader code

Gordon Surface
Results: Wing

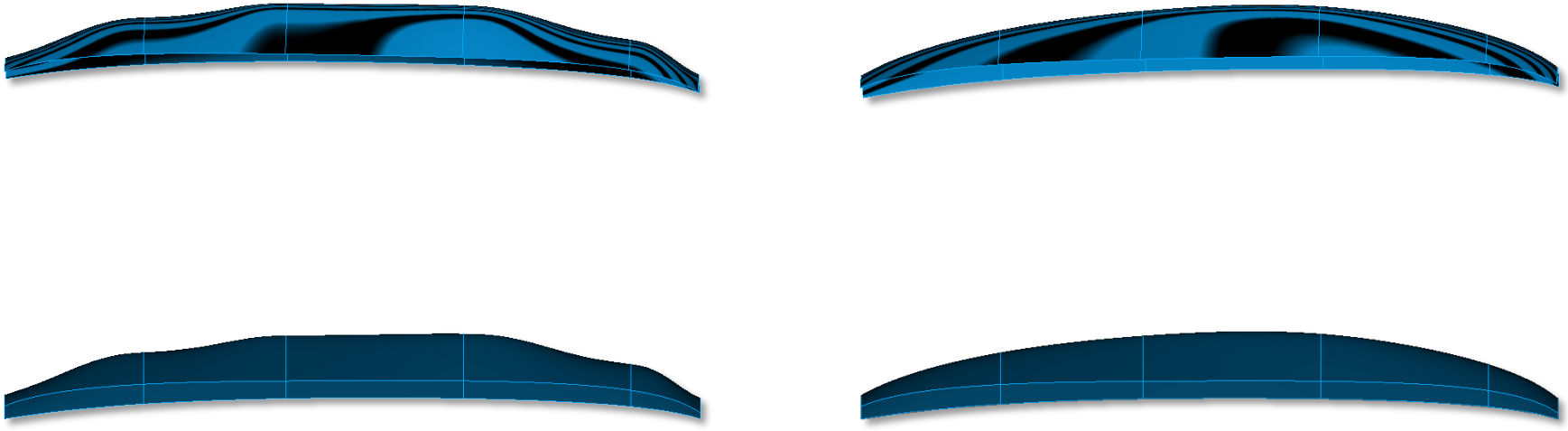


Old – Coons Style ☹️



New – Gordon Style 🙂

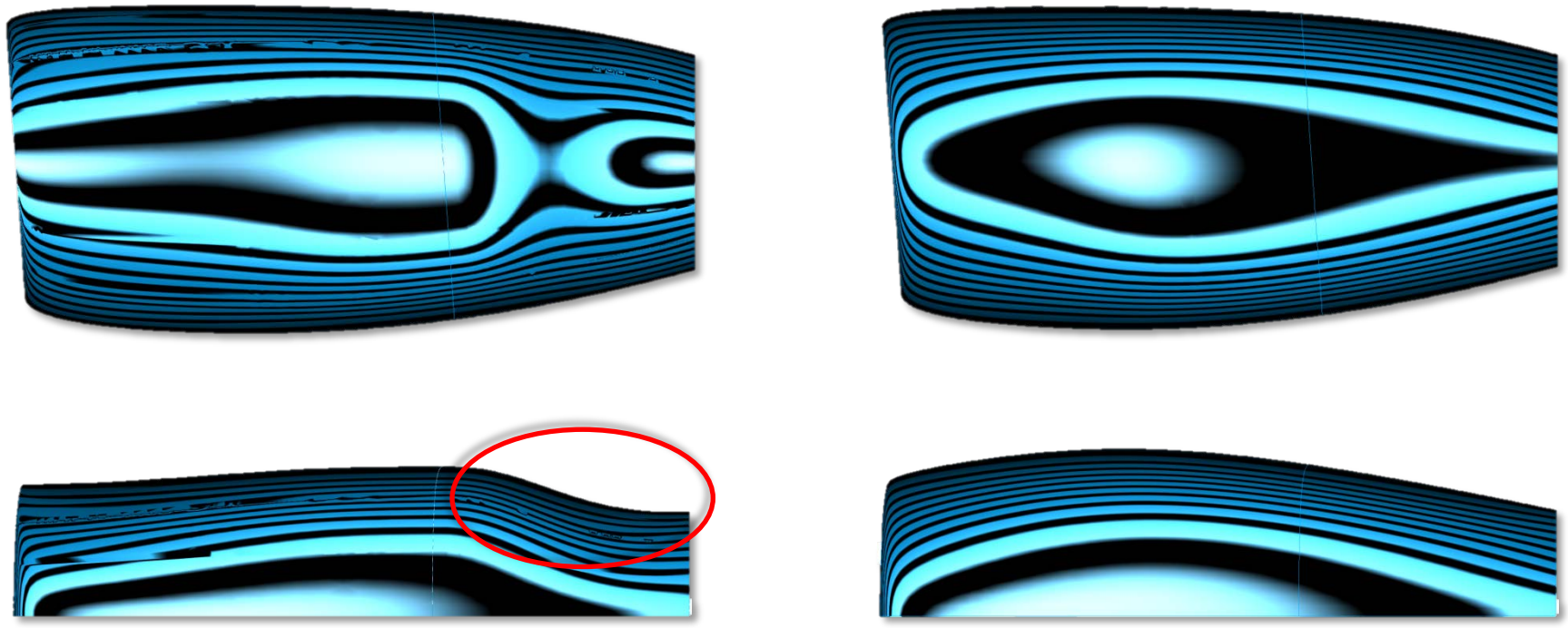
Gordon Surface
Results: Belly Fairing



Old – Coons Style ☹

New – Gordon Style 🙂

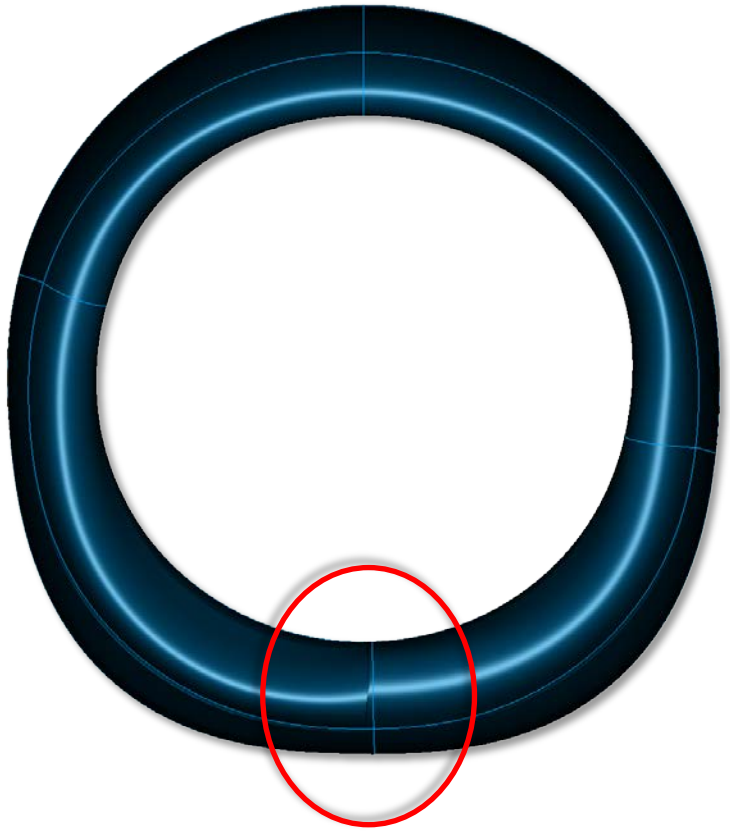
Gordon Surface
Results: Engine Cover



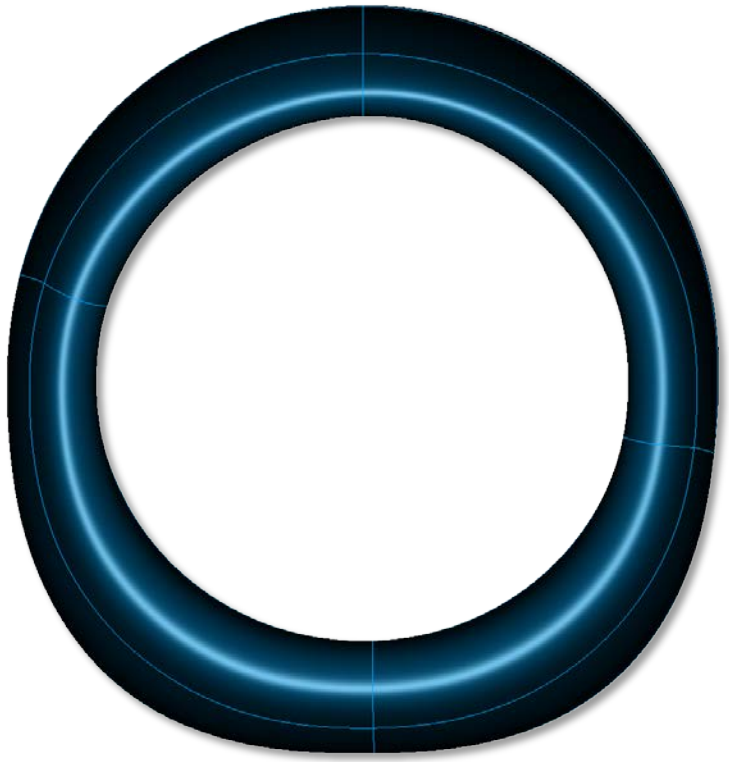
Old – Coons Style ☹

New – Gordon Style 🙂

Gordon Surface
Results: Engine Cover



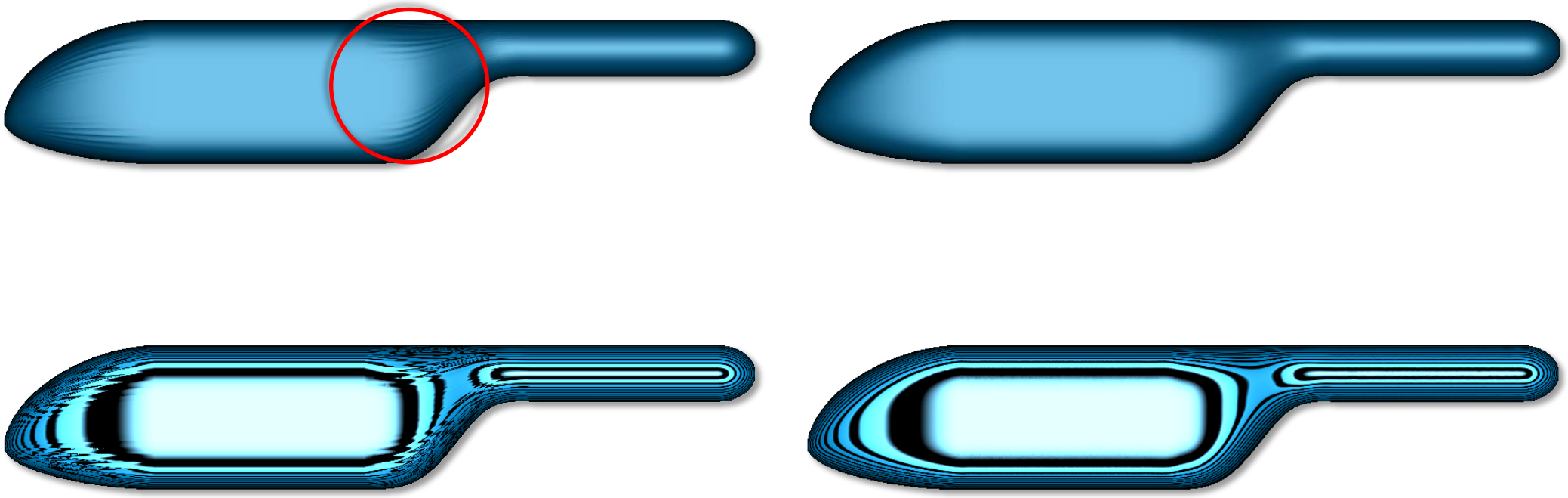
Old – Coons Style 😕



New – Gordon Style 🙂

Gordon Surface

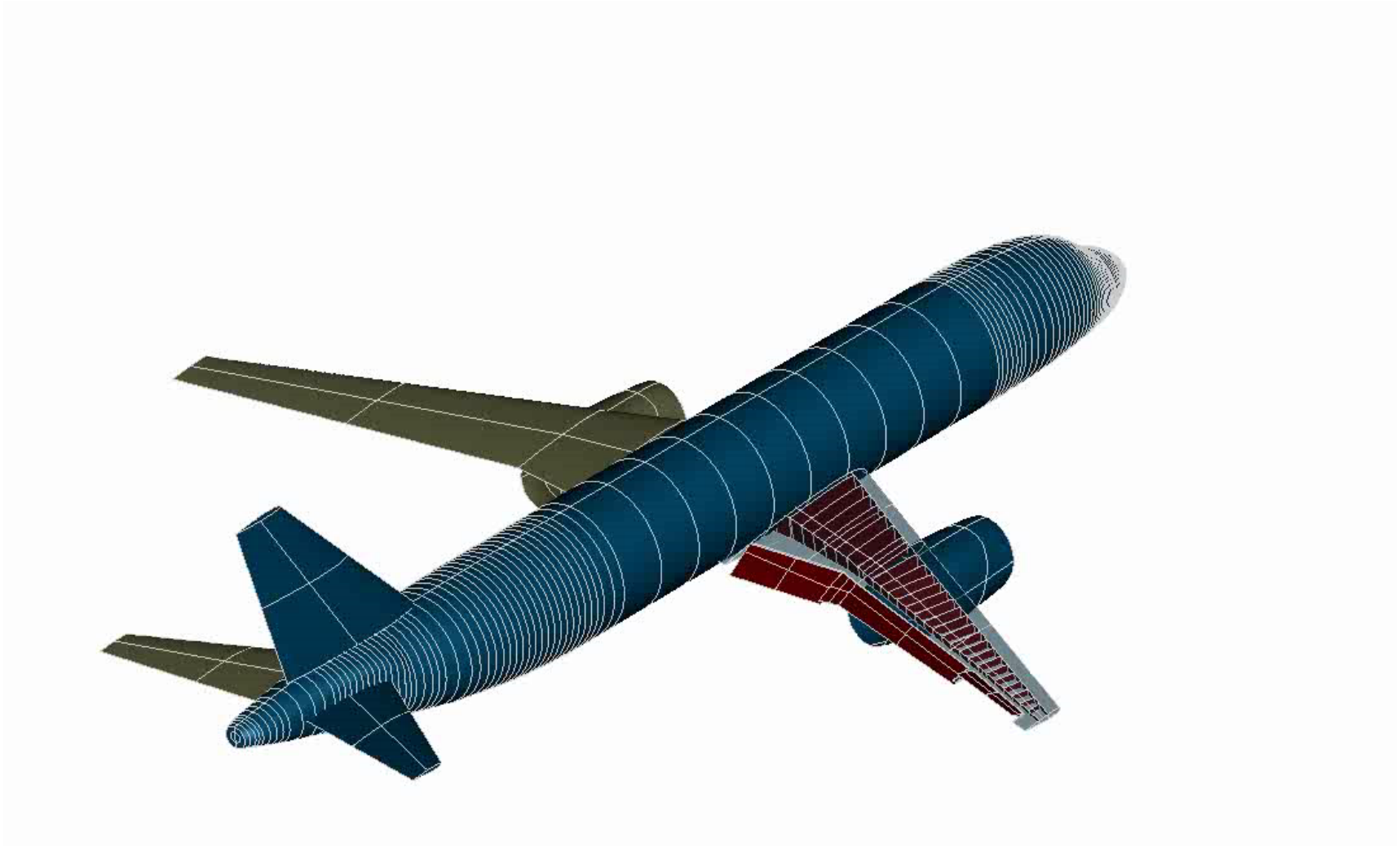
Results: Helicopter Body



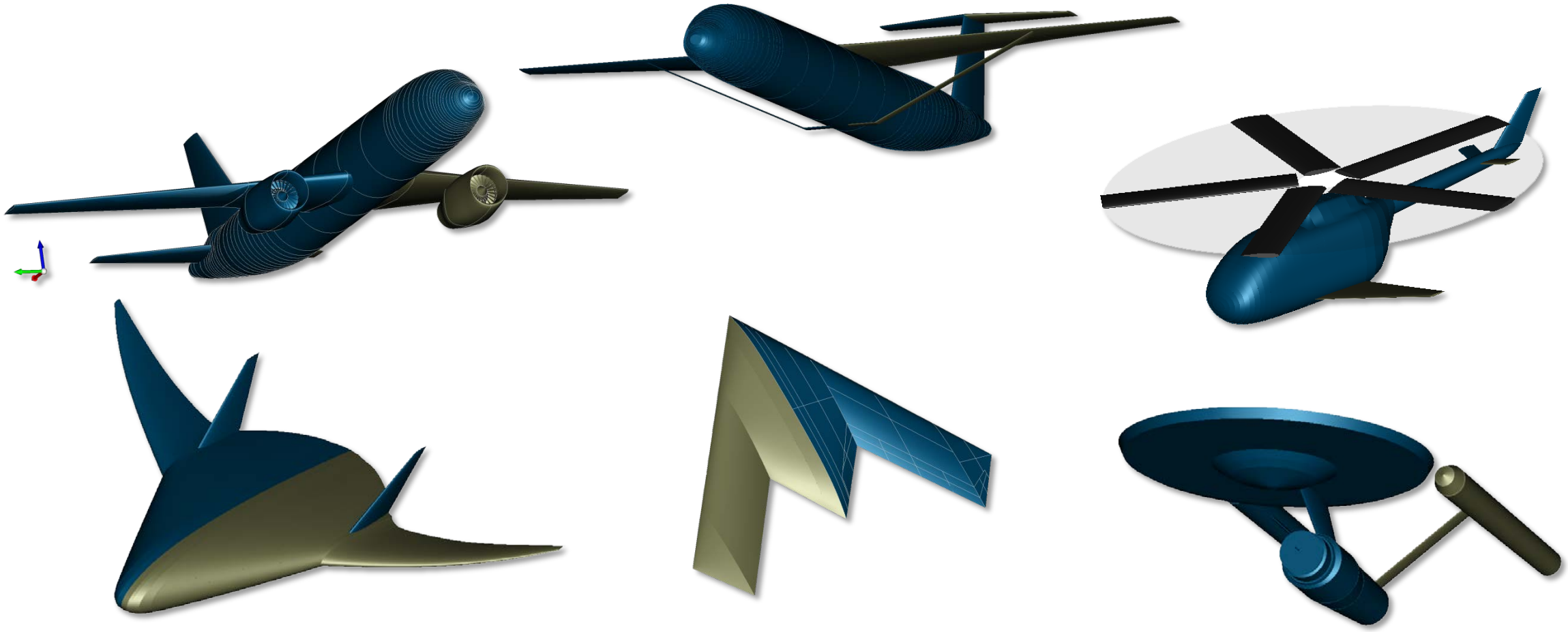
Old – Coons Style 😕

New – Gordon Style 🙂

Full aircraft model

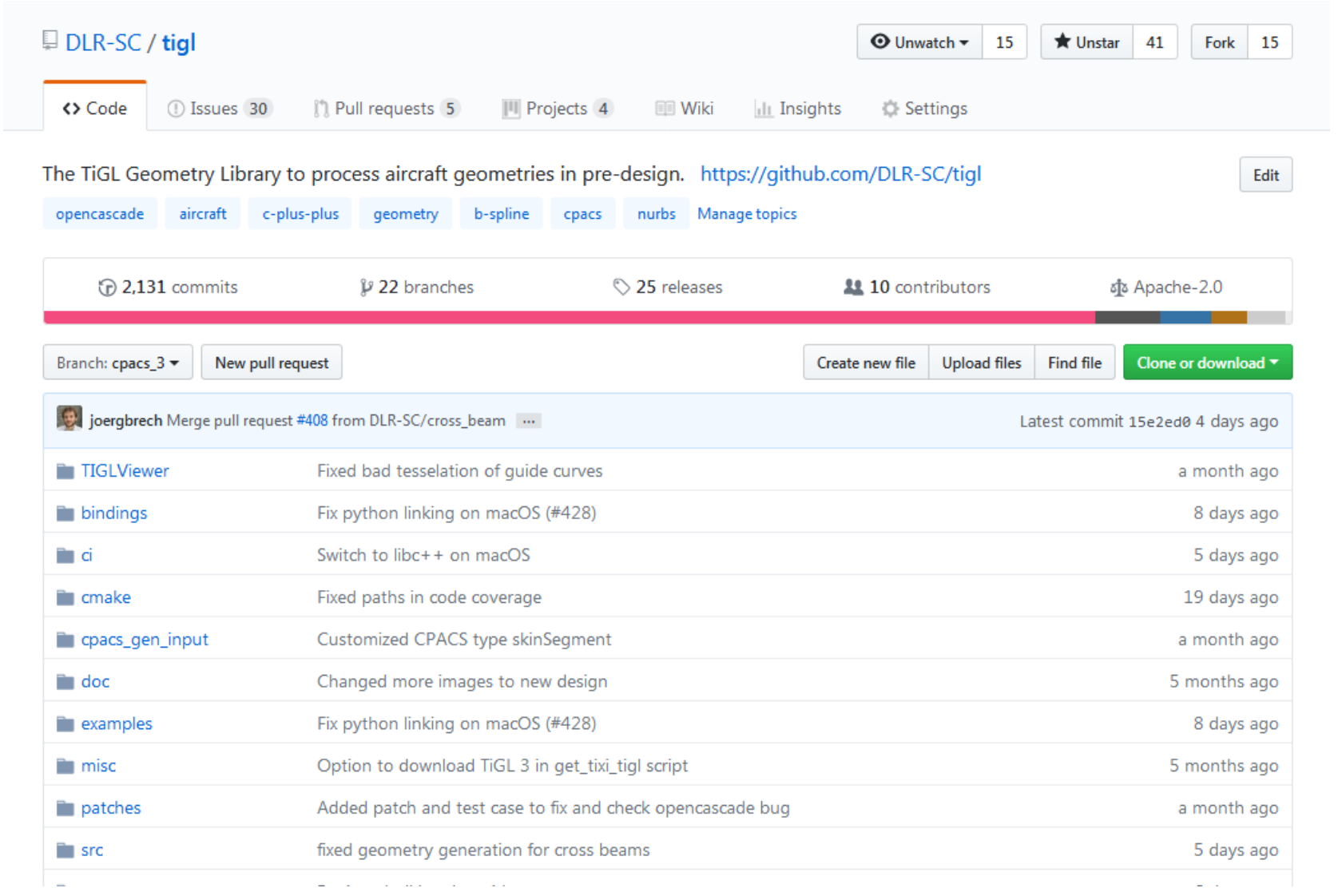


Possible aircraft designs



TiGL on GitHub

- Open Source, Apache-2.0 License
- ~ 120 kLOC
- ~43 kLOC auto-generated from CPACS XML schema
- https://github.com/DLR-SC/tigl



Conclusion

Summary

- TiGL is a library for geometric modeling of aircraft
- TiGL can be used for general purpose geometry modeling too
- Gordon Surfaces are a major building block for surfaces with high surface quality

Outlook

- Fuselage structure will come soon
- TiGL 3 Release probably in Q3 / 2018 (when CPACS 3 is finished)
- More aircraft specific geometries

Thanks to all TiGL Contributors

Sebastian Deinert (Airbus)
Bernhard Gruber (RISC)
Jonas Jepsen (DLR)
Jan Kleinert (DLR)
Philipp Kunze (DLR)
Roland Landertshammer (RISC)
Markus Litz (now Google)
Reinhold Maierl (Airbus)
Merlin Pelz (DLR)
Paul Putin (DLR)
Konstantin Rusch (DLR)
Tobias Stollenwerk (DLR)

Thank you for your attention!

martin.siggel@dlr.de